University of Zurich[UZH]

Zurich Open Repository and Archive
University of Zurich
University Library
Strickhofstrasse 39
CH-8057 Zurich
www.zora.uzh.ch

Year: 2012

A lightweight approach for product line scoping

Nöbauer, Markus ; Seyff, Norbert ; Groher, Iris ; Dhungana, Deepak

Abstract: Many organizations providing products with common features wish to take advantage of that similarity in order to reduce development and maintenance efforts. Their goal is to move from a single-system development paradigm towards a product line approach. However, this transition is not trivial and requires a systematic scoping phase to decide how the product line should be defined, i.e. what products and features should be included and thus developed for reuse. Currently available product line scoping approaches require huge upfront investments in the scoping phase, consuming a lot of time and resources. Our experience has shown that small and medium enterprises require a lightweight approach to decide how the existing products are related to each other so that their potential for reuse can be estimated more easily. In this paper we present a conceptual solution and early tool support enabling companies to semi-automatically identify similarity within existing product configurations.

DOI: https://doi.org/10.1109/SEAA.2012.81

Posted at the Zurich Open Repository and Archive, University of Zurich
ZORA URL: https://doi.org/10.5167/uzh-73247
Conference or Workshop Item
Accepted Version

Originally published at:
Nöbauer, Markus; Seyff, Norbert; Groher, Iris; Dhungana, Deepak (2012). A lightweight approach for product line scoping. In: EUROMICRO-SEAA, Cesme, Izmir, Turkey, 5 September 2012 - 8 September 2012. IEEE Computer Society, 105-108.
DOI: https://doi.org/10.1109/SEAA.2012.81

A Lightweight Approach For Product Line Scoping

Markus Nöbauer
InsideAx GmbH
A-4031 Linz, Austria
markus.noebauer@insideax.at

Norbert Seyff
University of Zurich
CH-8050 Zurich, Switzerland
seyff@ifi.uzh.ch

Iris Groher
Johannes Kepler University (JKU)
A-4020 Linz, Austria
iris.groher@jku.at

Deepak Dhungana
Siemens AG Austria
A-1211 Vienna, Austria
deepak.dhungana@siemens.com

Abstract— **Many organizations providing products with common features wish to take advantage of that similarity in order to reduce development and maintenance efforts. Their goal is to move from a single-system development paradigm towards a product line approach. However, this transition is not trivial and requires a systematic scoping phase to decide how the product line should be defined, i.e. what products and features should be included and thus developed for reuse. Currently available product line scoping approaches require huge upfront investments in the scoping phase, consuming a lot of time and resources. Our experience has shown that small and medium enterprises require a lightweight approach to decide how the existing products are related to each other so that their potential for reuse can be estimated more easily. In this paper we present a conceptual solution and early tool support enabling companies to semi-automatically identify similarity within existing product configurations.**

Keywords-product line scoping, product configuration

I. Introduction and Motivation

The advantages of adopting a product line (PL) approach through systematic reuse of artifacts have been demonstrated and proven to be successful in practice [2],[7],[8]. Nevertheless, many organizations still face a dilemma, when deciding whether or not to adopt this systematic reuse approach. A key issue is the decision about which existing products are potential candidates for a product line and thus promise an optimal return on investment. In product line engineering (PLE), these decisions are made in an early phase called *scoping*. Scoping identifies the commonality that members of the product line share and the ways in which they vary. In short, scoping answers the question "What products should be included in my product line and what features should they provide?"[10],[11], [5]. There is no general recipe for product line scoping as this process typically depends on the development practices within the organization, the architecture of the system, and the business context. In addition to software engineering, product line scoping is also related to marketing [6].

In this paper, we discuss the challenges of scoping from the perspective of small software companies already providing similar solutions for several customers. In practice, this is a common situation: because many specialized companies provide products for one or a small number of specific domains, several variants of the product often exist in parallel. Typically, such organizations wish to merge the efforts of individual products already delivered to customers to gain advantages from a product line approach. However, especially small and medium enterprises (SME) often cannot afford a huge up-front investment for building a platform [12]. Instead, they require an approach that supports a smooth transition from single system to product line development supporting reuse of existing legacy artifacts and an early payoff. A scoping process that fits the needs of these organizations is the first step in this direction. Researchers and practitioners have already presented ideas on how a scoping process can be customized to organizational and project constraints [3],[11]. However, existing scoping approaches require significant time and effort until a company can decide to actually start product line engineering. Scoping is often described as a long process involving multiple stakeholders who are able to make decisions from a different perspective (e.g. market analysts, software architects, and users). A fully-fledged long-term scoping approach includes many extensive tasks such as the definition of a product portfolio, product plans, and domain analysis. This can be a hurdle for smaller companies attempting to adopt a product line approach [12]. Therefore, a lightweight product line scoping approach is needed.

In this paper we present an approach that supports product line scoping based on existing product configurations without integrating product plans or portfolios into the decision process as existing scoping approaches suggest [5],[6],[11]. This paper is structured as follows: In Section 2 we discuss the research challenges of a lightweight scoping approach and present an approach that allows semi-automatic product line scoping based on existing product configurations. We describe a tool prototype which follows this approach. In Section 3 we discuss related work and conclude the paper in section 4.

II. Lightweight Product Line Scoping

A. Research Challenges

Within our work we have found that the introduction of software product lines is particularly challenging for smaller companies. These companies usually have limited resources to introduce product lines and have difficulties to afford platform development and maintenance. Furthermore, their products often rely heavily on rapidly evolving third-party components

Proc. 38th Euromicro Conf. on Software Engineering
and Advanced Applications (SEAA 2012)
Cesme, Izmir, Turkey, Sept. 2012. 105-108.

(e.g. in the ERP domain where partner companies customize products from large software vendors for customers). This creates a highly volatile environment. Although companies could basically benefit from product line approaches, they often refuse this option as the costs could exceed the benefits. For example, cases where the time needed to introduce the product line outruns the lifetime of the underlying 3rd party components.

Another major issue is the people involved in product line scoping activities. A lack of objective methods to investigate the reuse potential within a company's software products leaves them with their own personal opinions and estimates. We experienced that people then tend to judge the situation based on what they think of the reuse potential of their software. Different backgrounds, views, etc. might lead to misjudgments and discussions which are not based on hard facts but potentially false opinions.

The goal of our current work is to realize a lightweight product line scoping approach; it would permit the identification of key information on the reuse potential of existing products in a fast and efficient way. In order to achieve this, we pursued the following key steps: The problem analysis and the definition of the research goals was followed by a more detailed literature review to identify existing work relevant to our research. In a next step, we developed a conceptual solution which enables companies to identify similarity within their products based on existing product configurations. We then implemented a first tool prototype which semi-automatically identifies reuse potential.

B. Scoping based on Existing Configurations

Our research focuses on settings where product configurations are used to tailor the products to individual customer needs and provide the requested features. We target domains where products are mainly customized by configurations and where companies already have a range of products delivered to customers. Typically, a large set of configuration parameters are offered that allow tailoring the system to the individual needs of the customer. We focus on products where individual configurations are derived from a certain configuration schema. For such settings we developed a conceptual solution which allows to process and compare these configurations. The comparisons are used to identify the degree of similarity between products. The pre-requisite for our approach is the availability of a set of product configurations reflecting the solutions developed for different customers.

Our approach comprises the following steps. These steps need to be conducted by an experienced domain expert such as the product manager or a group of people who have the required insights:

1) Define Analysis' Domain Scope: In a first step domain experts select a set of products for comparison. The selection criteria can be rather simple, for example, a company can decide to compare all products delivered in a certain timeframe or all products in a certain domain e.g. retail stores. This selection is refined based on market considerations and the companies strategy for further development.

2) Define Analysis' Product Scope: We assume that target products are configured for different customers by setting values on predefined configuration parameters and activating or deactivating product features. Therefore, in a second step, a domain expert has to identify scoping-relevant configuration parameters. The identification of relevant configuration parameters is considered to be the key to the similarity analysis between products. In our appraoch, product wide configurations are initially selected to scope the analysis (e.g. pricing configurations shall be part of the analysis). In most cases it is necessary to tailor this scope and to exclude customer-specific settings that are different in all configurations. Domain experts may remove these parts from the configurations before automated processing starts.

3) Organize Data for Analysis: Configuration parameters are often not a good measure to investigate the resuse potential of a software product. The settings can be of different granularity and have various effects on the product. For example, a configuration parameter may be used for turning on/off functionality (e.g. perform a credit check before sending an order confirmation) while others may represent some more fine granular adjustments (e.g. use a comprehensive table view or a simple list view). For this reason, in a third step, we recommend to group settings to features rather than performing a direct comparison of settings. Moreover, this enables domain experts to weight features (e.g. UI settings may be not as relevant as security features). Finally, the features repesenting a particular functionality are grouped accordingly in hirarchical order reflecting the system architecture.

4) Calculate Similarity: The next step is to calculate the similarity of selected products. This is done by comparing the individual products' feature configuration. Here, the domain expert has to answer the question: Do all selected products require the same order processing checks? The percentage of similar features is calculated pairwise for all selected products. Based on individual comparisons, a report representing a degree-of-similarity analysis is generated. This report can be tailored to view only certain aspects of the compared products or narrow the scope based on market considerations (e.g. only retail store systems are delivered to customers in Eastern Europe).

5) Discuss Findings: Feedback from developers and product managers responsible for each product is important as these can reveal insider-information to guide a company to setup a PL approach in future customer projects (e.g. using a decision model). In a fifth step, the results of the similarity analysis are presented to domain experts which may reveal a similarity threshold value: it can be used for future analysis as guideline for including/excluding products/features from a product line. For example, experts might come to the conclusion that a similarity higher than 60% is a good indicator for inclusion in a PL. State-of-the-art literature also discusses similarity thresholds which indicate the reuse potential of software products [4]

6) Derive Default Configuration: Based on the identified scope of the product line, generation of default system configurations for certain domains can be automated. In a last step, the scope is reduced to the commonalities between products which will support sales and system engineers to deploy or demonstrate a customized product for a customer. Default configurations can also build the platform for the conceived product line.

C. *Tool Support*

We have implemented a first prototpye tool that supports our lightweight scoping process. It can be used to analyse the similarity between existing customized products. In the following, we describe the necessary steps to calculate the similarity between different product configurations.

1) Load Configuration Schema: In an initial step the general configuration schema is loaded from a meta-data definition file, including all possible settings. This step also automates the feature grouping as described in the last section (step 3). The tool creates a hierarchy of product configurations, including features and their configuration settings. The tool also imports the configuration settings' data types.

2) Define Similarity Ranges: In many cases it is required to allow a more "fuzzy" similarity comparison, where values are not exactly the same, but within a predefined similarity range. We therefore introduced optional similarity ranges for configuration settings. The tool supports basic mathematical operations in a way that allows domain experts to provide more complex boundaries. For example, {0} + ({0}/100)*10 represents an 10 percentage upper bound, where {0} is the original value. Furthermore, some settings may not be relevant for similarity calculation and therefore should not be included in the analysis. For example, systems using a database to store configurations often provide audit fields such as Modified By and CreatedDateTime. Although these fields are part of the configuration schema, its very unlikely to identify any similarity in other product configurations. Therefore, we provide an option to exclude such settings from the similarity calculation.

3) Load Product Configuration: The tool manages multiple customized product configurations. Domain experts can optionally add meta-data like legal entities and the customer's industry branches. We distinguish between legal entities because they often represent companies working in different branches of industry. For example, a product that is customized to support retail might contain one legal entity that actually sells goods to customers. The same product can also contain a second legal entity that, for example, supports the first one by providing finance services for customers. Although both legal entities use the same customized product, their configurations differ. Furthermore, we support the comparison between selected legal entities' configurations from different products (e.g. compare Customers X finance service with Customers Y finance service).

4) Setup Similarity Analysis: The product configurations for analysis are selected in this step. The domain expert needs to select two or more legal entities and sets the scope of the analysis (e.g. Sales, Purchase). One legal entity is selected as master-configuration. The tool calculates the similarity ranges for every configuration setting based on the value in the master-configuration. For example, if freight charges are 100€ in the master-configuration and the similarity range is defined +/- 25€ the tool calculates 75€ - 125€ as valid similarity intervall.

5) Perform Similarity Analysis: The tool calculates a similarity value by comparing the configuration settings per feature. Only if all product configuration settings for a feature are similar (equal or within the similarity range) the feature is identified as similar. The overall similarity value is calculated as the percentage for similar features. This value can serve as a basis for further discussion with domain experts, if a product line approach is feasible. The tool also supports the comparison between more than two product configurations at once. However, in such a comparison, one highly divergent configuration may significantly reduce the calculated similarity. Therefore, we also foresee a pairwise comparison to identify the outlier. This outlier may be excluded in a future product line.

Figure 1 shows how the tool presents the calculated similarity of selected customized products. Each line shows the pairwise similarity regarding typical industry activities such as Sales and Purchase. The value on the right hand side shows the similarity value in percent.

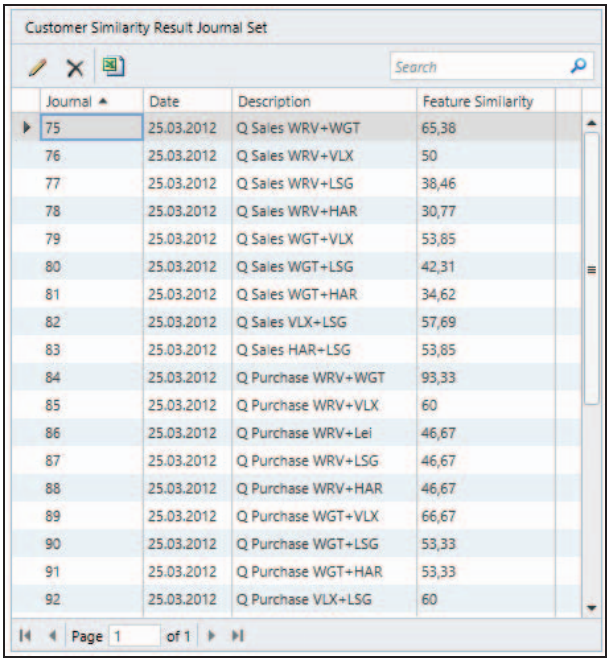

Figure 1 Similarity Analysis Results

III. Related Work

Approaches dealing with scoping in the context of software product line engineering and transitioning to software product lines are related to our approach.

A. *Software Product Line Scoping*

In [11], PuLSE-BEAT is introduced, a tool for supporting the product line scoping approach called PuLSE-Eco presented in [10]. To identify the optimal scope, a product map is used; it is a matrix with the product characteristics (features) on one axis and the products on the other axis. The product characteristics are elicited from stakeholders, existing systems and the product plan. In our approach, product characteristics (we call them feature definitions) are derived from existing systems. Our tool creates a product map based on imported feature definitions and configuration schemas. In PuLSE-Eco, the benefit analysis step decides what to develop for reuse and what not. Benefit functions describe the benefit of having a certain characteristic inside the scope and typically range from 0 to 1. PuLSE-BEAT supports three different scopes (0.5

means possible candidate, 0.6 means likely candidate, and 0.75 means strongly recommended candidate). We analyze the similarity of the different products (configurations) to decide what should be inside the scope and therefore developed for reuse.

In [6]**Fehler! Verweisquelle konnte nicht gefunden werden.** different levels of scoping are identified, namely product portfolio scoping, domain scoping and asset scoping. Our approach falls into the domain scoping category where the domain that is important to the product line and provides sufficient reuse potential is bounded. The scoping process consists of three phases: product line mapping, domain potential assessment and reuse infrastructure scoping. Our approach focuses on the first two phases. Product line mapping identifies products and features and builds a product feature matrix. Domain potential assessment evaluates sub-domains of the application domain for their reuse potential. Our approach also focuses on identifying domains with a high reuse potential.

In [1] RiPLE-SC, an agile product line scoping process, is presented. The process consists of different phases that are performed iteratively, namely pre-scoping, domain scoping, product scoping and asset scoping. Our approach fits into the domain and product scoping phases which aim to determine the domains and features with high potential. The scoping process presented does not explicitly support a commonality analysis of existing products as our approach provides. As our approach is lightweight with a high degree of automation it could easily be integrated into the RiPLE-SC process.

B. Transitioning to Software Product Lines

In [6] we present an approach and toolkit that lowers the up-front costs and adoption barriers that typically come along with the introduction of a software product line approach. One of the three different adoption models presented, the extractive approach, analyses existing products and extracts common and varying parts into a product line. Applying such an approach requires customized products to be available that have a significant amount of commonality. It is suggested that the high-payoff systems should be extracted initially. This fits well with our approach that analyses existing products to find common parts that can be extracted in a second step.

In [4] an approach is presented that analyzes the source code of multiple variants for commonalities to support migration towards a product line. The reuse potential of system parts is assessed by using occurrence matrices. Instead of a pair-wise comparison of existing variants, the authors propose to describe the similarity between a set of variants in a matrix. The matrix contains the different elements of the variants that are compared, the variants and the occurrence of the elements in the variants. The similarity rate is categorized as core (element occurs in all variants), shared (element occurs in some variants), and unique (element occurs in only one variant).

IV. Conclusions And Further Work

In our current work we focus on product line scoping approaches which can be used by SMEs in order to get fast feedback on their software's reuse potential. These companies often cannot afford to apply state-of-the-art scoping approaches. Several issues limit their application. For instance, some SME companies have no long-term control over their software artifacts. This highly volatile environment creates a need for lightweight approaches. In this paper, we presented a light-weight and semi-automatic approach. In addition to a conceptual solution we present a tool prototype which supports product line scoping based on comparing individual product feature configurations.

Future work will include extending the tool functionality to not only support product line scoping and similarity analysis but to also support the identification of highly similar parts across different customized products. We consider this to be an important next step to improve the support for SMEs to set up a managed reuse of assets.

As well as tool improvements we also plan extensive evaluations. In particular we plan an evaluation involving domain experts at InsideAx. In addition, we will make the tool available to other companies whose products are customized using configurations.

Acknowledgment

The research conducted was in part funded by the Austrian Research Promotion Agency (FFG, Project Nr.: 2025605).

References

[1] M. Balbino, E. S. de Almeida, and S. Meira, "An Agile Scoping Process for Software Product Lines", 23rd International Conference on Software Engineering and Knowledge Engineering (SEKE), Miami Beach, USA, pp. 717-722, 2011.

[2] P. Clements and L. M. Northrop, "Software Product Lines:Practices and Patterns", Addison Wesley, 2007.

[3] J.-M. DeBaud and K. Schmid, "A systematic approach to derive the scope of software product lines", 21st International Conference on Software Engineering (ICSE), Los Angeles, California, USA, pp. 34-43, 1999.

[4] S. Duszynski, J. Knodel, and M. Becker, "Analyzing the Source Code of Multiple Software Variants for Reuse Potential", 18th Working Conference on Reverse Engineering (WCRE), Limerick, Ireland, pp. 303-307, 2011.

[5] I. John and M. Eisenbarth, "A Decade of Scoping – A Survey", 13th International Software Product Line Conference (SPLC), San Francisco, California, USA, pp. 31-40, 2009.

[6] I. John, J. Knodel, T. Lehner, and D. Muthig, "A Practical Guide to Product Line Scoping", 10th International Software Product Line Conference (SPLC), Baltimore, Maryland, USA, pp. 3-12, 2006.

[7] K. C. Kang, V. Sugumaran, and S. Park, "Applied Software Product Line Engineering", Auerbach Pubn, 2009.

[8] F. J. van der Linden, K. Schmid, and E. Rommes, "Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering", Springer Berlin Heidelberg, 2007.

[9] K. Schmid, "Scoping Software Product Lines: An Analysis of an Emerging Technology", 1st Conference on Software Product Lines (SPLC), Denver, Colorado, USA, pp. 513-532, 2000.

[10] K. Schmid, "A comprehensive product line scoping approach and its validation", 22rd International Conference on Software Engineering (ICSE 2002), Orlando, Florida, pp. 593-603, 2002.

[11] K. Schmid and M. Schank, "PuLSE-BEAT – A Decision Support Tool for Scoping Product Lines", 3rd International Workshop on Software Architectures for Product Families (SAPF), Las Palmas de Gran Canaria, Spain, pp. 65-75, 2000.

[12] M. Verlage and T. Kiesgen, "Five years of product line engineering in a small company", 27th International Conference on Software Engineering (ICSE), St. Louis, MO, USA, pp. 534-543, 2005.